

Adaptable Program Synthesis

PROBLEM

Program synthesis can automatically generate high quality code to solve problems in complex domains from clear, modular specifications. For well-understood problem domains and target platforms, automatic synthesis yields very substantial benefits in code development time and quality. For less well understood domains or target platforms, it is generally necessary to manually edit synthesized code, reducing the benefits of synthesis.

TECHNOLOGY

AutoFilter is a program synthesis system developed in the Robust Software Engineering group at NASA Ames. *AutoFilter* generates Kalman filter code from specifications of state estimation problems, e.g. for GN&C. We adapted *AutoFilter* to synthesize *components* for state estimation, and implemented a compiler for a C-like language interfaced to the synthesized code. We used the modified system in a case study, synthesizing code for a complex FIDO rover estimator which combines several Kalman filters with non Kalman filter code. When tested in simulation, state estimates agreed with those from hand-written code (graph has 6 lines, but 3 from synthesized code overlap 3 from hand-written code).

SOLUTION

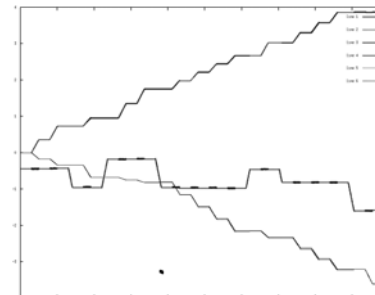
We extended a synthesis system to allow users to augment the mathematical model with a specification of additional code necessary for the application. The extension enables automatic synthesis of code for a wide range of real-world problems. In the domain of state estimation, for example, this includes:

- arbitrary mode changes,
- combinations of Kalman filters with non-Kalman filter code.

specification

Mathematical model +
application-specific code

automatic
synthesis



accurate
estimates

Explanation of Accomplishment

- **POC:** Julian Richardson (RIACS/USRA, RSE Group, Code TI, julianr@email.arc.nasa.gov)
- **Work funded by:** RSO Program Synthesis.
- **Background:** Domain-specific program synthesis, e.g. as implemented in *AutoBayes* (data analysis) and *AutoFilter* (state estimation – Kalman filters) can generate high quality code to solve problems in complex domains from clear, modular specifications. Commercial program synthesizers/generators typically yield 3-5x reductions in development time, 2x reductions in errors, and reduced maintenance costs. In a NASA context we can offer even greater gains, for example the complete elimination of entire classes of bugs through certification. Discussion with JPL GN&C experts indicated that applications in the GN&C domain frequently require modifications to the standard Kalman filter algorithms, for example to handle bad sensor data, or perform a part of the algorithm in a non-standard way. These can be achieved by manual modifications to synthesized code, but this reduces the benefits arising from synthesis. For example, whenever the specification is changed and the code resynthesized, these edits would need to be redone – time consuming and error-prone work.
- **Accomplishment:** We have extended the *AutoFilter* synthesis system to allow users to specify application-specific code which is interleaved with the domain-specific synthesized code. Synthesis is automatic. This significantly extends the scope of the synthesis system. We adapted *AutoFilter* so that for each specification, it generates a *library of components* for implementing that specification, rather than a single synthesized program. We designed and implemented a compiler for an imperative language, which allows the user to combine components from the synthesized library with their own code, and so implement algorithms and architectures not covered by the synthesis system. We demonstrated the effectiveness of the approach by synthesizing a complex state estimator for the FIDO rover. The estimator combines two Kalman filters with non Kalman filter code. The synthesized code was compared with a state estimator written by Dr. Ed Wilson (Discovery and Systems Health). The results of the two implementations agreed.
- **Benefits:** This accomplishment significantly expands the scope of automatic synthesis, potentially extending its benefits (e.g. cost reduction, quality improvement, sustainability) to greater portions of NASA mission applications.
- **Future Work:** We expect that the certification capabilities developed in the Robust Software Engineering Group can be very effectively leveraged to ensure that the combination of hand-written and synthesized code has desirable correctness properties (e.g. array bounds safety, variable initialization, maintains symmetry of key matrices).